From reproducibly analyzing Fitbit activity data to visualizing results.
JHSPH Biostat Qualifying Exam 2013 Take Home re-take

Abstract

We analyzed the number of steps taken by an individual for a period of two months in 2012 that was recorded using a Fitbit[2] device. Personal measurement devices have been on the rise and there is demand for new statistical methods to deal with this type of data. We estimated the average number of steps taken per day, inferred the average activity pattern within a day, and determined whether there was a difference in the activity pattern between weekdays and weekends. To do so, we explored the possibility of predicting missing observations. Finally, we built a Shiny[4] application that allows anyone to upload their own data and analyze it with the methods implemented in this project.

This project is completely reproducible and all the code has been compiled in the `fitbitR` R package[1].

Introduction

Fitbit[2] is one of the popular devices on the market for collecting personal data such as the number of steps taken. Their devices can record activity data each minute and through the Fitbit API you can download your own activity data in different window intervals; for example every 5 minutes. Some enthusiasts have analyzed their own data[5] but questions remain such as what is the average number of steps taken per day, average activity patterns (within a day), and whether these patterns are different between weekends and weekdays. This project answers these questions and further allows users to analyze their own data through an application built with Shiny[4].

In particular, we have data from a single individual for a period of two months. As shown in Figure 1 (top) out of the 61 days, 8 are missing (shown in gray) with no obvious missing pattern. This plot also allows us to check for any week patterns. For example, this individual had an irregular week in mid October as (s)he was not as active at 8 am compared to other weeks. Visualizing the data in 24 hour circular clock makes it much easier to notice the hours of the day when someone is active, but has the caveat of not showing missing observations. In particular, this individual is regularly active from 6 am to 7pm on weekdays as shown in Figure 1 (bottom). Similarly, this person is mostly active from 8 am to 9pm on weekends. Furthermore, Fridays seem to fall out of the usual weekday activity pattern.

Overall, there is strong indication of a difference in activity patterns between weekends and weekdays. Figure 1 (both) shows that the high activity peaks are more consistent on weekdays although they do change by date, versus the high peaks on the weekends which are more variable.

Methods and Results

Predicting missing observations

Four methods for predicting the missing observations have been implemented in `fitbitR`[1]. The first one, *overall-mean*, simply replaces the observations by the overall mean. The second one, *means*, replaces the missing observations by the mean from similar observations: those from the same interval and day of the week. The third one, *lm*, fits a linear regression model with a 10 degree of freedom natural spline on the Interval covariate, date and day of the week (dow) using

the following model for the number of steps Y_i, $i = 1, 2, \ldots, n$:

$$Y_i = \beta_0 + \sum_{j=1}^{10} \beta_j \text{ns}_j(\text{interval}_i) + \beta_{11}\text{date}_i + \beta_{12}\text{dow}_i + \epsilon_i$$

The predicted values are truncated at 0 for any negative predictions (if any). The fourth method, *poisson*, fits a Poisson GLM using the same covariate structure as the *lm* method (without ϵ_i and with the appropriate link function). The four methods, among others[1], were evaluated by training on 70% of the non-missing data and predicting on the remaining 30%. The error measure used is the root mean square prediction error (RMSPE).

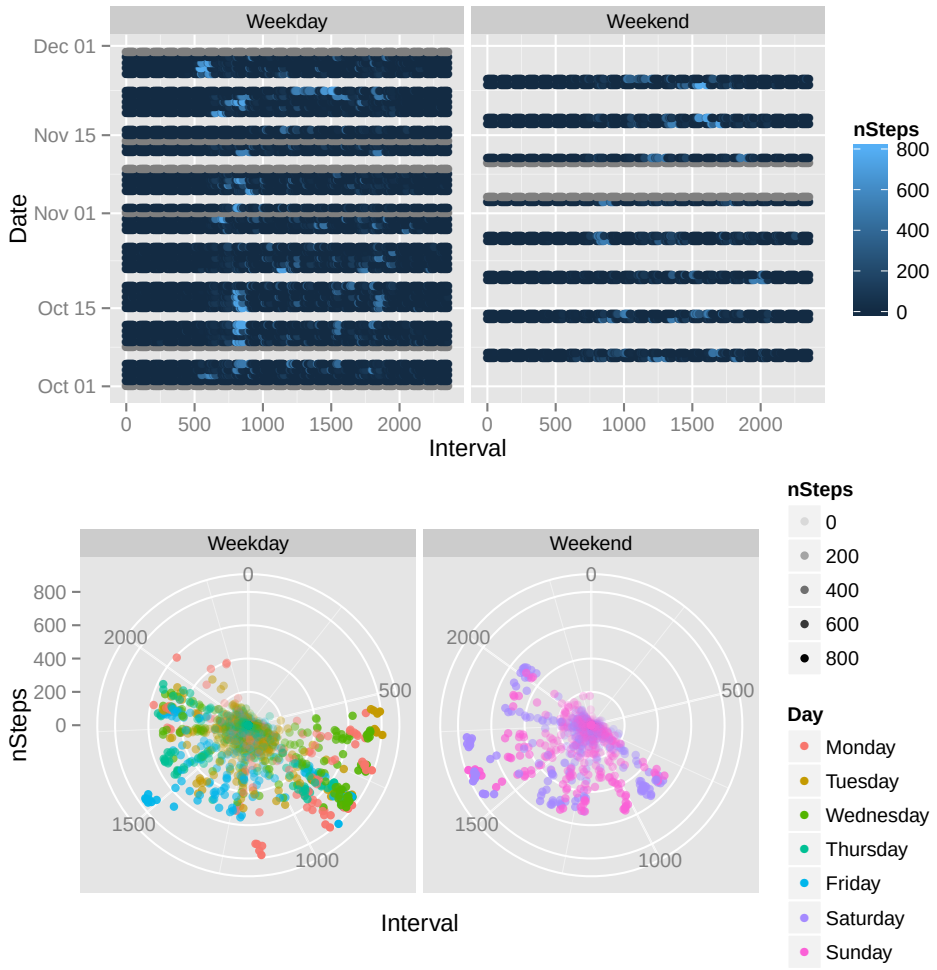

Figure 1: Exploratory plots of the number of steps (nSteps) for a specific individual along a two month period with data binned by 5 minute intervals. Top plot shows the data by Interval and Date separated by whether the day is a weekday or not. The activity peaks (light blue) are more consistent for weekdays while weekends seem more variable. Missing observations (gray) are clearly visible only in this plot. The bottom plot shows the data in a in 24 hour clock where we can clearly notice that this individual is regularly mostly active from 6 am to 7 pm on weekends and 8 am to 9 pm on weekends.

[1]Check `reproduceAnalysis("pred")` for more information.

overall-mean and *means* are straight-forward methods to predict the missing observations, but they heavily assume that the observations are missing at random and that there is a consistent pattern across similar observations (for *means*). *overall-mean* was used as a benchmark for poor-prediction.

lm was used despite the non-normality of the data due to it's robustness. We were not expecting a good result from this method, but it did out-perform *overall-mean* and *means*. *poisson* was used because the data are counts –despite overdispersion problems– and the flexibility of using natural splines. In both *lm* and *poisson*, 10 degrees of freedom on the natural splines worked better than using more.

We found that the *lm* and *poisson* methods were practically tied but did not improve the RMSPE by much: *overall-mean* 102.9 (SE 3.689), *means* 109.9 (SE 2.898), *lm* 99.7 (SE 3.283), and *poisson* 99.89 (SE 3.223). Surprisingly *means* was outperformed by *overall-mean*, which could be an indicator that the error measure is sensible to outliers and a more robust one should be used.

Average number of steps taken per day

Once the data is binned by day, the straightforward method to estimate the average number of steps taken by day is using the sample mean. The problem with this estimator is that it ignores the correlation in the number of steps taken between day i and day $i + 1$. To deal with this structure, we fitted several ARIMA models[3] on both the number of steps by interval and the data binned by day before choosing to use an ARIMA(3, 0, 3) model with the interval data (not binned).

The naive method has lower standard errors than those from the ARIMA(3, 0, 3) model, presumably because the naive model underestimates the actual number of steps taken per day by assuming that the measurements are independent. The results are shown in Table 1 for the **original** data and the four prediction methods. Note that there were no significant differences (t-tests for the difference in sample mean[2]) between the original data and the predicted sets.

	Estimate	SE	95% CI:L	95% CI:U
original	10766.90	813.73	9171.90	12361.90
lm	10851.70	711.69	9456.71	12246.68
poisson	10828.01	715.11	9426.33	12229.70
means	10822.89	719.74	9412.13	12233.65
overall-mean	10766.19	697.35	9399.31	12133.07

Table 1: Estimated average number of steps taken per day using an ARIMA(3, 0, 3) model on the interval data (not binned). The estimate, standard error and 95 percent confidence intervals (based on the t-distribution) are shown for fitting the ARIMA(3, 0, 3) model to the original data with missing observations as well as the completed data using the four prediction methods previously described.

Average activity pattern

To infer the average activity pattern we compared a naive versus a model based approach. The naive approach is to take the mean of the observations grouped by interval. This results in a highly variable estimate of the average activity pattern as shown in the appendix (Figure 3). It is thus important to smooth the data in order to have an interpretable average activity pattern.

For this purpose we fitted a General Additive Model (GAM) using the quasipoisson family because the data is a set of counts with high overdispersion. We used cubic spline basis for the

[2]Check `reproduceAnalysis("Q1")` for more information.

interval time of day (transformed to integer scale). The cubic spline basis are useful for generating a smooth curve that can be easier to interpret. Figure 2 (top) shows the overall activity pattern independent of the day of the week. From Figure 2 (top) we can interpret that this individual regularly wakes up at 6 am, is highly active from 7 to 9 am peaking at 8 am, keeps a steady level of activity from 10 am to 4pm, has slightly increased activity levels from 5 to 6pm, and then decreases until 9 pm after which (s)he is rarely active and is presumably sleeping.

This result was invariant to whether the **original** data or one with predicted values was used[3].

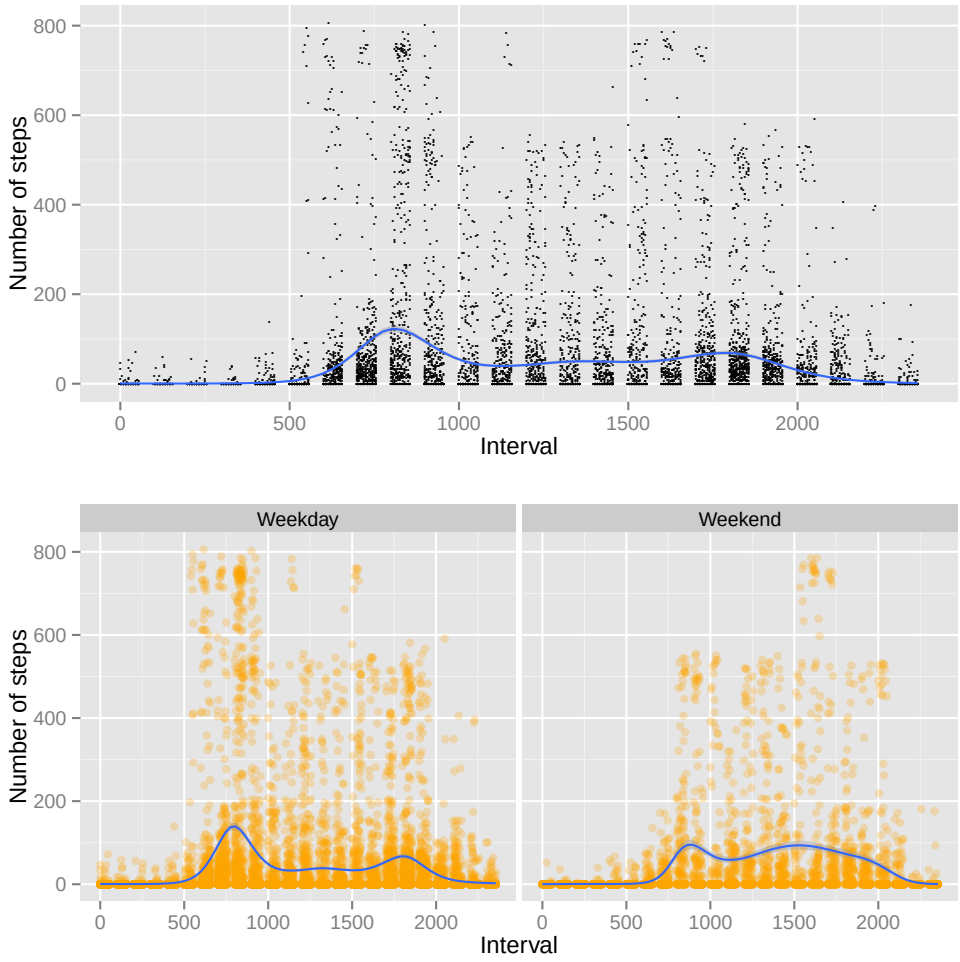

Figure 2: Average activity pattern over time (within a day). Top plot uses all the data while the bottom plot separates the data by whether it's a weekend or a weekday. Blue curves are GAM models fitted for the quasipoisson family using cubic spline basis on the interval time of day.

Weekdays vs weekends

To determine whether there is a difference in activity patterns between weekdays and weekends, we fitted a GAM model as described previously but with an additional indicator covariate *Weekend* differentiating weekdays (0) from weekends (1). The estimate of the *Weekend* coefficient –in the log scale– is 0.1981 95% CI:(0.138, 0.258) based on the t-distribution with 15263 degrees of freedom. It

[3]Most easily noted when using `fitbitShine()` in tab Q2 and changing the prediction method.

can be interpreted as a 21.9 percent change associated with a weekend vs weekday. Since 0 is not included in the 95% CI, there is a significant difference in the activity patterns between weekdays and weekends.

In Figure 2 (bottom) we can notice how this individual has two activity peaks during weekdays (7-9 am and 5-6 3pm) with a steady plateau in between them (10 am to 4pm). On weekends, this individual gets active later in the morning and keeps a rather similar activity level during the day (except for brunch time: 10 am to noon) and eventually stops being active later on the day (around 10 pm) compared to weekdays.

This result was invariant to whether the **original** data or one with predicted values was used[4].

Conclusions

This individual takes an average of 10767 steps per day 95% CI:(9172, 12362), gets active early in the morning before stabilizing until his/her work is probably over at 5pm and goes home. During weekends, (s)he gets active later during the day and overall keeps a higher level of activity on weekends.

The methods developed in this project should prove helpful to other individuals who want to analyze their Fitbit[2] activity data and visualize the results using the Shiny[4] web application implemented in `fitbitR`[1]. The Shiny application includes the prediction methods although they did not affect the results in this current analysis but could potentially do so depending on the data set.

References

[1] L. Collado-Torres. *fitbitR: JHSPH Biostat qualifying exam 2013 take home re-take by L. Collado-Torres*. R package version 0.3. 2013. URL: `https://github.com/lcolladotor/fitbitR`.

[2] Fitbit Inc. *fitbit*. URL: `http://fitbit.com` (visited on 06/06/2013).

[3] R. Hyndman and Y. Khandakar. "Automatic time series forecasting: The forecast package for R". In: *Journal of Statistical Software* (2008).

[4] RStudio and Inc. *shiny: Web Application Framework for R*. R package version 0.7.0. 2013. URL: `http://CRAN.R-project.org/package=shiny`.

[5] J. T. Rubin. *One Full Year of FitBit Pedometer Data*. URL: `http://www.jamierubin.net/2013/03/20/one-full-year-of-fitbit-pedometer-data-part-1-a-look-back/` (visited on 06/04/2013).

[4] Most easily noted when using `fitbitShine()` in tab Q3 and changing the prediction method.

A Average activity pattern: mean method

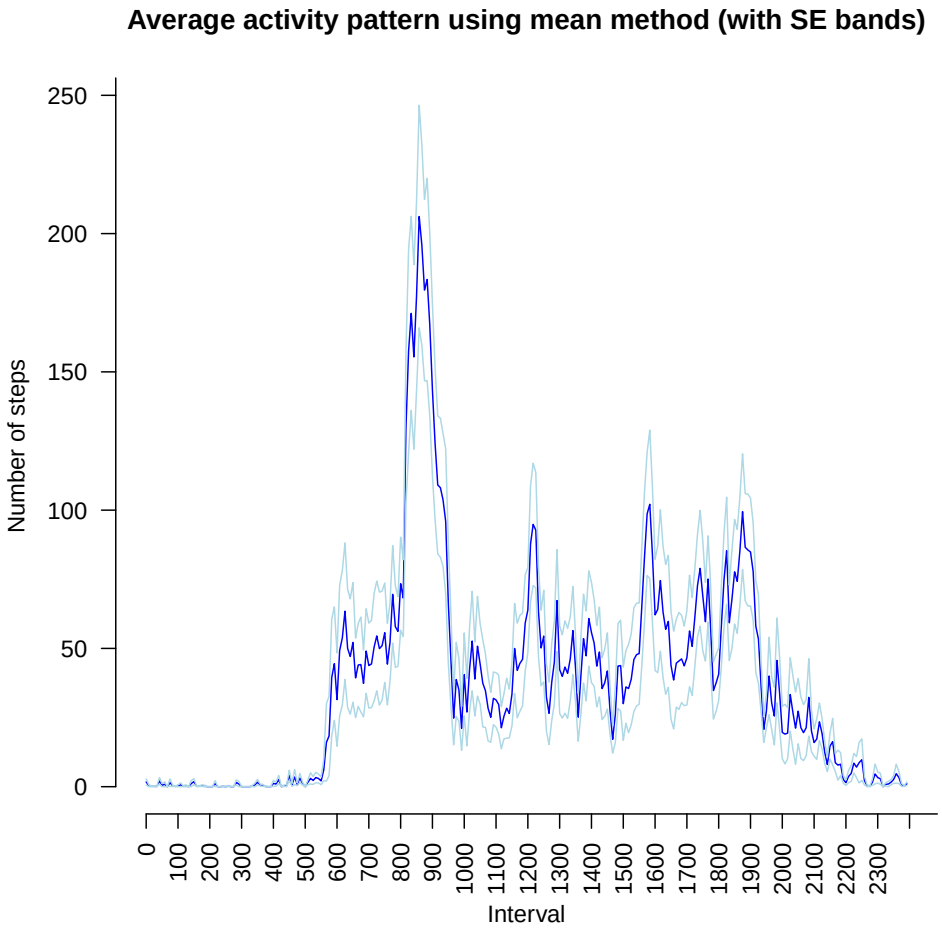

Figure 3: Average activity pattern over time (within a day) using the naive mean method.

B Running the Shiny application

To run the Shiny application, you just have to run the following commands from R.

```
library(fitbitR)
fitbitShine()
```

C Reproducibility

Please check `https://github.com/lcolladotor/fitbitR` for details on how to install the *fitbitR* package (which passes R CMD check!) and reproduce the results including this report.

Note that `reproduceAnalysis("all")` takes around 4 minutes to complete.